TIZEN BASED REMOTE CONTROLLER CAR USING RASPBERRY PI2

Pintu Kumar (pintu.k@samsung.com, pintu_agarwal@yahoo.com)

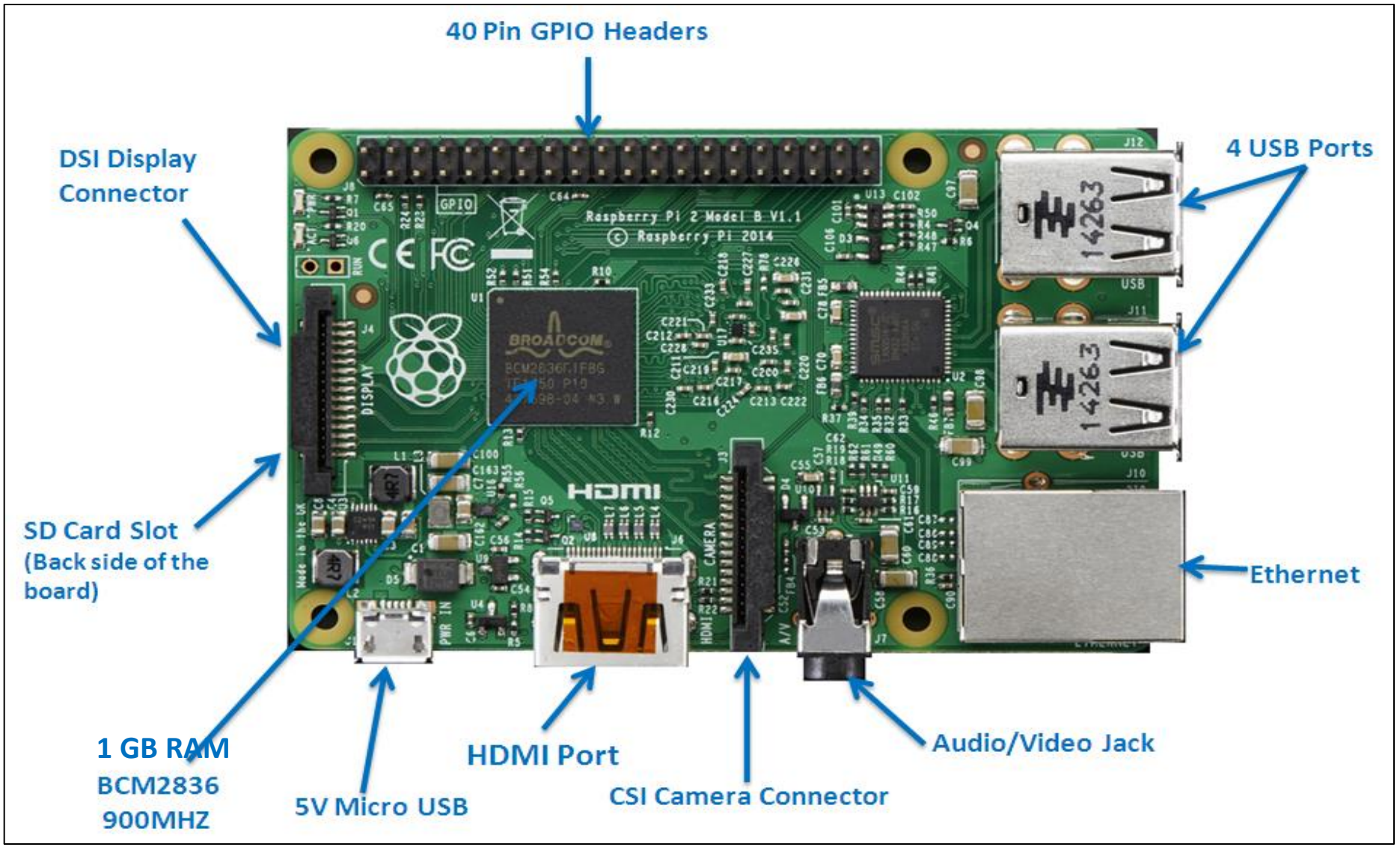
Samsung Research India – Bangalore : Tizen Kernel/BSP Team

CONTENT

INTRODUCTION

- This talk is about designing a remote controller robot (toy car) using the raspberry pi2 hardware, pi2 Linux Kernel and Tizen OS as platform.
- In this presentation, first we will see how to replace and boot Tizen OS on Raspberry Pi using the pre-built Tizen images. Then we will see how to setup Bluetooth, Wi-Fi on Tizen and finally see how to control a robot remotely using Tizen smart phone application.

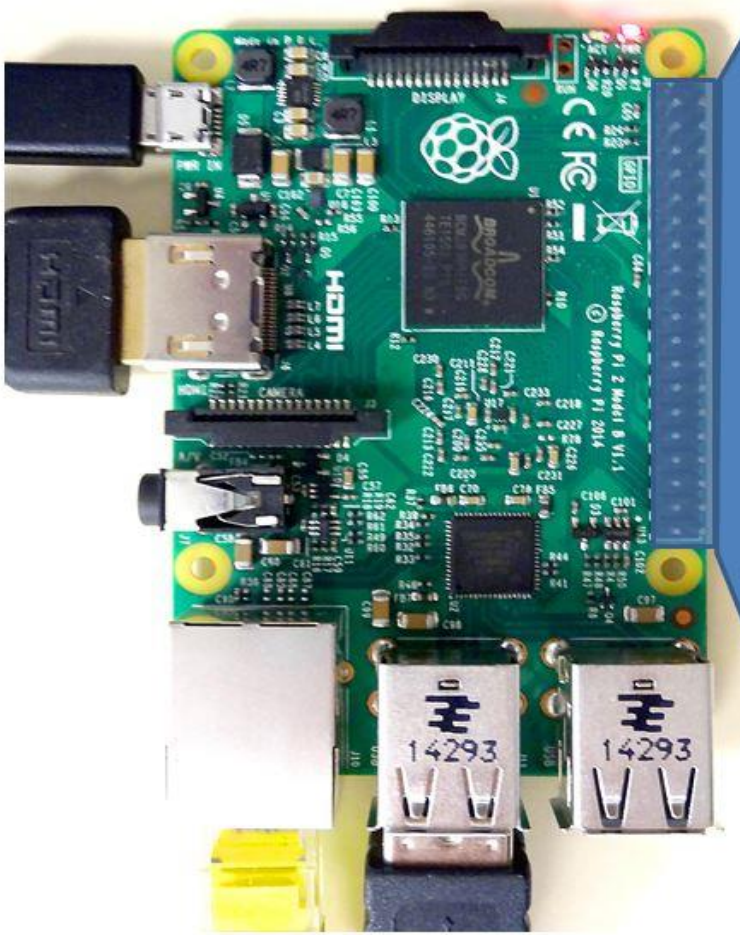
RASPBERRY PI2 - OVERVIEW



Raspberry PI2 Features

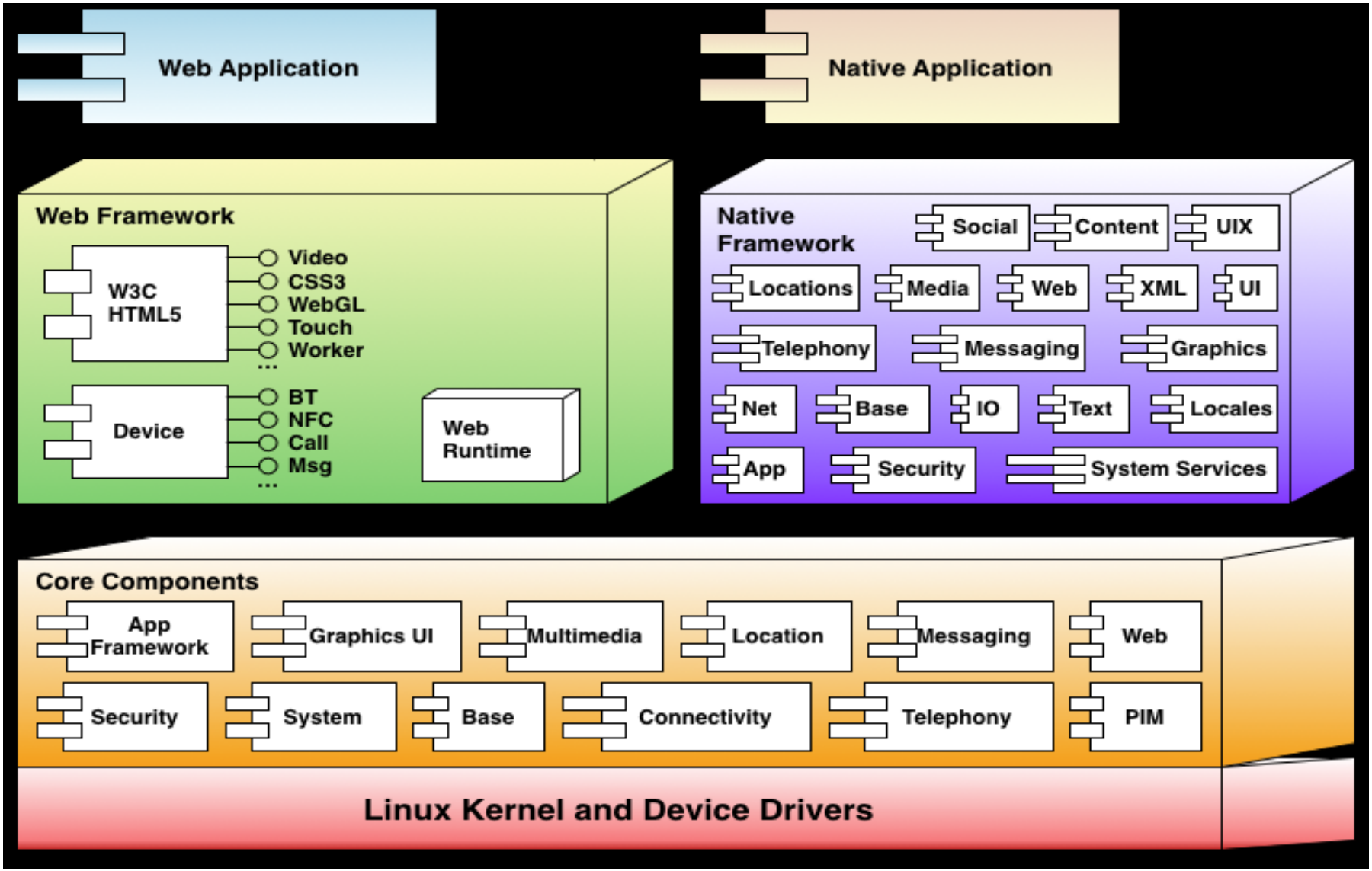
- Broadcom BCM2836 900MHz Quad Core ARM Cortex-A7 CPU
- 1GB RAM
- 4 USB ports
- 40 GPIO pins
- Full HDMI port
- Ethernet port
- Combined 3.5mm audio jack and composite video
- Camera interface (CSI)
- Display interface (DSI)
- Micro SD card slot
- Video Core IV 3D graphics core

PI2 GPIO Pins

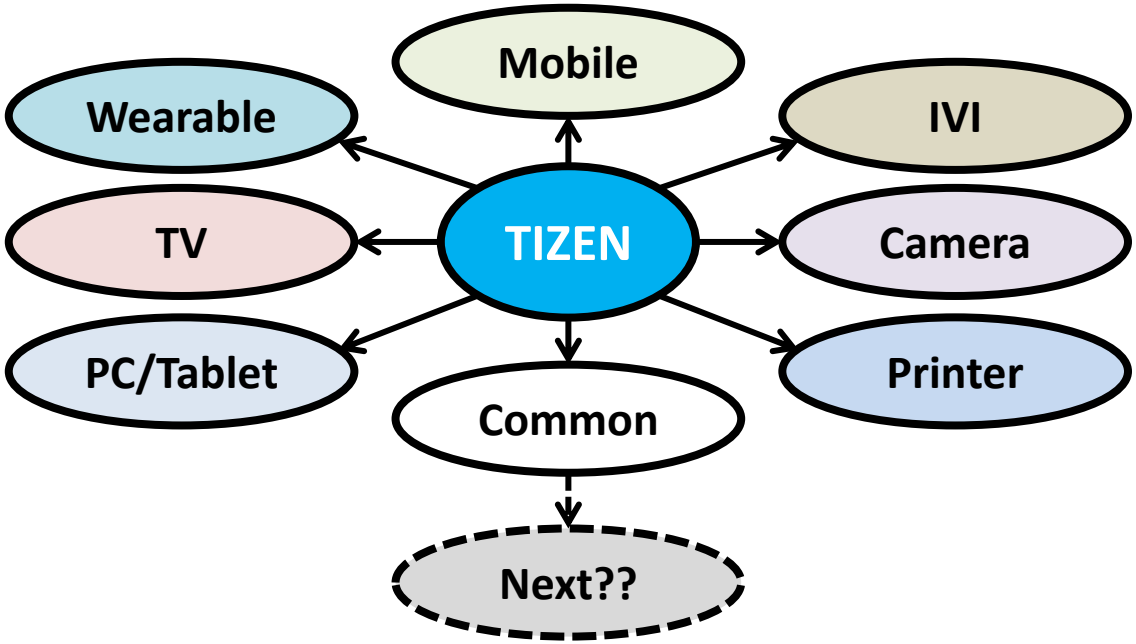


		Physical Pins			
GPIO#	2nd func	pin#	pin#	2nd func	GPIO#
N/A	+3V3	1	2	+5V	N/A
GPIO2	SDA1 (I2C)	3	4	+5V	N/A
GPIO3	SCL1 (I2C)	5	6	GND	N/A
GPIO4	GCLK	7	8	TXD0 (UART)	GPIO14
N/A	GND	9	10	RXD0 (UART)	GPIO15
GPIO17	GEN0	11	12	GEN1	GPIO18
GPIO27	GEN2	13	14	GND	N/A
GPIO22	GEN3	15	16	GEN4	GPIO23
N/A	+3V3	17	18	GEN5	GPIO24
GPIO10	MOSI (SPI)	19	20	GND	N/A
GPIO9	MISO (SPI)	21	22	GEN6	GPIO25
GPIO11	SCLK (SPI)	23	24	CE0_N (SPI)	GPIO8
N/A	GND	25	26	CE1_N (SPI)	GPIO7
EEPROM	ID_SD	27	28	ID_SC	EEPROM
GPIO5	N/A	29	30	GND	N/A
GPIO6	N/A	31	32	-	GPIO12
GPIO13	N/A	33	34	GND	N/A
GPIO19	N/A	35	36	N/A	GPIO16
GPIO26	N/A	37	38	N/A	GPIO20
N/A	GND	39	40	N/A	GPIO21

TIZEN OVERVIEW



TIZEN Profiles



- TIZEN is the OS of everything.
- Tizen is a multi-device OS which can support many types of profiles.
- The current profile that are supported are:
  - Mobile
  - Wearable
  - IVI
  - Common
- The new profiles can be easily derived using the minimal common profile.

TIZEN Features

- Tizen is truly open source. Almost all components are based on open source packages.
- Uses mainline Linux Kernel
- Uses systemd for booting
- Uses dbus for IPC communication
- Uses DRM/X11/Wayland for Display & Graphics
- Uses Gstreamer for multimedia framework
- Uses SMACK for platform security
- Uses EFL (Enlightenment Foundation Libraries) for UI framework
- Provides SDB (Smart Development Bridge) for developers.
- Uses HTML5 for WebApps development
- And many more….

HARDWARE COMPONENTS

- Raspberry pi 2 hardware
- Linux PC – Ubuntu 14.04
- Micro SD Card (8 GB)
- Robot Chassis platform (with 2 DC motors, 2 wheels, 1 Castor wheels)
- L293D Driver Board (1 number)
- USB Power Bank (1 number)
- AA size batteries (8 numbers, 12V)
- Battery holder/case (1 number)
- Wi-Fi USB Dongle (1 number)
- Bluetooth USB dongle (1 number)
- USB Web Cam (1 number)
- A Monitor Screen (for Display purpose)
- HDMI Cable (1 number)
- USB keyboard & mouse
- A Tizen Smart Phone with Tizen 2.4
- Screws, Blots, Spacer, jumper wires etc.

SOFTWARE COMPONENTS

- Raspberry Pi – NOOBS image
- Tizen 3.0 common pre-built images (alternatively Tizen pi2 pre-built image).
- Raspberry Pi Linux Kernel 4.1.16
- GCC ARM tool chain (arm-linux-gnueabi-gcc)
- Tizen Yocto setup (Or, Tizen GBS Build setup)
- Tizen 2.4 SDK software
- Ubuntu 14.04

Raspberry PI Download

- Download Raspberry pi software from:
  - https://www.raspberrypi.org/downloads/
- Extract it and install it on the SD card.
- Boot the raspberry pi using this SD card.
- Install the Raspbian OS and boot it till desktop.
- At this time verify that all functionalities are working fine on Raspberry pi image.

TIZEN Images Download

- **Download Tizen images from:**
  - https://download.tizen.org/
- **Choose any one type of image from the below repo.**

Name	Last modified	Size	Description
docker/	2015-09-11 04:36	-	
iris/	2014-09-03 03:31	-	
lecture/	2015-07-19 23:02	-	
live/	2015-04-13 13:33	-	
misc/	2015-03-04 14:35	-	
prerelease/	2014-07-02 13:31	-	
releases/	2015-10-28 01:19	-	
sdk/	2016-02-03 09:12	-	
services/	2015-04-29 11:41	-	
snapshots/	2015-11-26 22:22	-	
tct/	2015-10-28 05:35	-	
tools/	2015-09-02 06:48	-	

- **If you want to try latest release mobile profile, you can use this:**
  - **https://download.tizen.org/releases/2.4/2.4-mobile/tizen-2.4-mobile_20151030.1/images/**
- **If you want to use common profile, you can use this:**
  - **https://download.tizen.org/snapshots/tizen/common/latest/images/**

TIZEN PI2 Images

- **Tizen raspberry pi 2 pre-built images:**
  - **https://files.s-osg.org/tizen-on-rpi2/**

Index of /tizen-on-rpi2

Name	Last modified	Size
Parent Directory		-
README.txt	12-Aug-2015 10:42	1.2K
bblayers.conf	18-Aug-2015 02:24	1.3K
ex.tizen.rpi-sdimg.2015-07-15	15-Jul-2015 13:18	704M
local.conf	06-Aug-2015 15:41	4.5K
local.conf.3d_accel_vc	03-Sep-2015 07:10	3.3K
md5sum.txt	12-Aug-2015 10:58	57
rpm/	14-Aug-2015 10:24	-
tizen-common-core-image-crosswalk-dev-raspberrypi2-20150806223520.rootfs.rpi-sdimg	06-Aug-2015 18:18	1.2G
tizen-common-core-image-crosswalk-dev-raspberrypi2-20150811204400.rootfs.rpi-sdimg	11-Aug-2015 14:15	1.2G
tizen-common-core-image-crosswalk-dev-raspberrypi2.rpi-sdimg-2015-08-06	06-Aug-2015 18:18	1.2G
tizen-common-core-image-crosswalk-dev-raspberrypi2.rpi-sdimg-2015-08-11	11-Aug-2015 14:15	1.2G
tizen-common-core-image-crosswalk-dev-raspberrypi2.rpi-sdimg-2015-08-14	14-Aug-2015 10:25	1.4G
tizen-common-core-image-crosswalk-dev.sdimg-2015-08-06	06-Aug-2015 16:33	1.2G
tizen.rpi-sdimg.2015-04-22	22-Apr-2015 14:18	596M
tizen.rpi-sdimg.2015-07-14	14-Jul-2015 08:13	620M
tizen.rpi-sdimg.2015-08-04	04-Aug-2015 08:22	856M
tizen.rpi-sdimg.2015-08-12	12-Aug-2015 10:35	728M
tizen.rpi-sdimg.EXPERIMENTAL	15-Jul-2015 13:18	704M
tizen.rpi-sdimg.LATEST	12-Aug-2015 10:35	728M

TIZEN Build Setup

- **GBS Build System:**
  - https://source.tizen.org/es/documentation/reference/git-build-system?langredirect=1

- **YOCTO Build System:**
  - https://wiki.tizen.org/wiki/Build_Tizen_with_Yocto_Project
  - https://wiki.tizen.org/wiki/Tizen_on_Yocto_Project

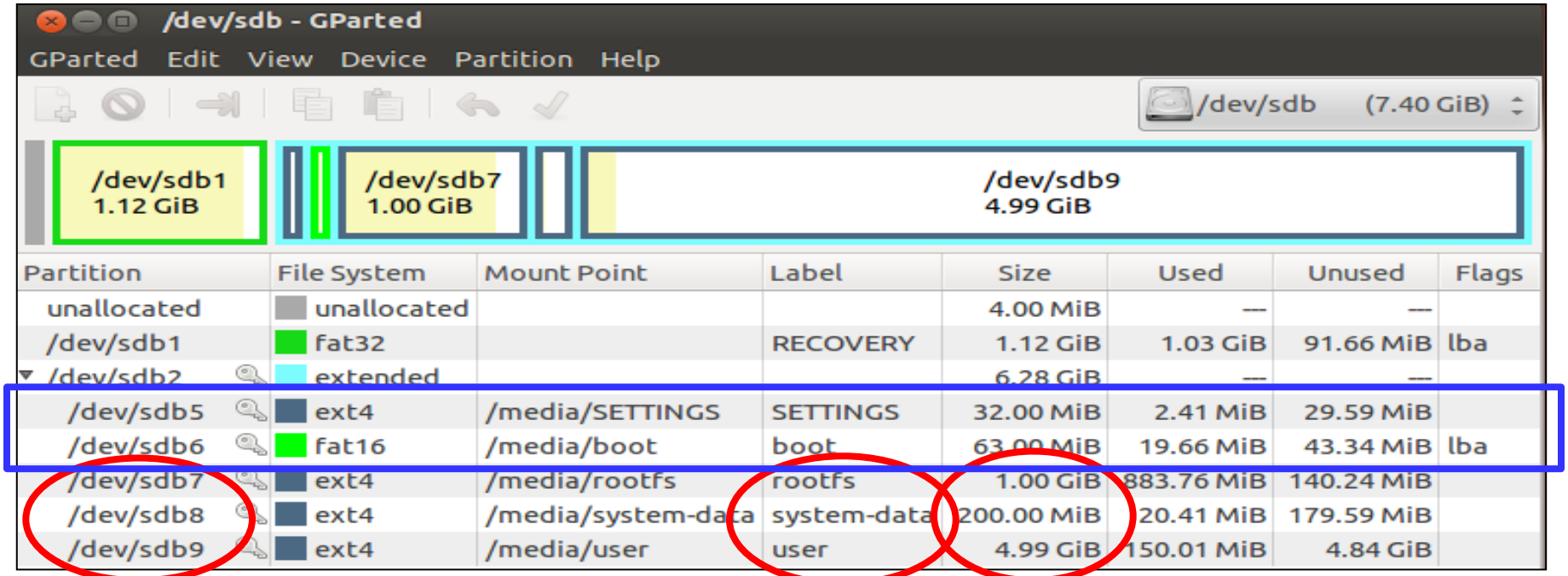
Tizen 3.0 common pre-built rpms:

https://download.tizen.org/snapshots/tizen/common/latest/repos/arm-wayland/packages/armv7l/

TIZEN Partition Creation

- Extract the Tizen common image on the Linux PC. It will contain 3 images:
  - rootfs.img (root file system)
  - system-data.img (system partition: /opt)
  - user.img (user partition: /opt/usr)
- Now, check the size of each image using the “du –h” command.
  - # du –h *.img
    - 864M rootfs.img
    - 49M system-data.img
    - 97M user.img

- Use Gparted on Ubuntu to create new partitions for Tizen images, on the SD card.
- First erase the raspberry pi OS root partition. Do not disturb the SETTINGS and boot partitions.
- Then create the new partitions as follows:



Partition	File System	Mount Point	Label	Size	Used	Unused	Flags
unallocated	unallocated			4.00 MiB	---	---	
/dev/sdb1	fat32		RECOVERY	1.12 GiB	1.03 GiB	91.66 MiB	lba
/dev/sdb2	extended			6.28 GiB	---	---	
/dev/sdb5	ext4	/media/SETTINGS	SETTINGS	32.00 MiB	2.41 MiB	29.59 MiB	
/dev/sdb6	fat16	/media/boot	boot	63.00 MiB	19.66 MiB	43.34 MiB	lba
/dev/sdb7	ext4	/media/rootfs	rootfs	1.00 GiB	883.76 MiB	140.24 MiB	
/dev/sdb8	ext4	/media/system-data	system-data	200.00 MiB	20.41 MiB	179.59 MiB	
/dev/sdb9	ext4	/media/user	user	4.99 GiB	150.01 MiB	4.84 GiB	

- **Make sure to run resize2fs command to resize all the partitions.**
  - **sudo resize2fs /dev/sdb7 [rootfs partition]**
  - **sudo resize2fs /dev/sdb8 [system-data partition]**
  - **sudo resize2fs /dev/sdb9 [user partition]**
- **Now, use "dd" commands in Linux to write the actual Tizen images to the respective partitions on SD card.**
  - sudo dd if=**rootfs.img** of=/dev/**sdb7** bs=4M
  - sudo dd if=**system-data.img** of=/dev/**sdb8** bs=4M
  - sudo dd if=**user.img** of=/dev/**sdb9** bs=4M

- **Then, remount the SD card on the Linux PC.**
- **Now, using the "df –h" command, we should be able to see all the partitions as follows:**

```
/dev/sdb6       63M   20M   44M  32% /media/boot
/dev/sdb8      196M   17M  180M   9% /media/system-data
/dev/sdb5       31M  1.4M   28M   5% /media/SETTINGS
/dev/sdb9      5.0G   72M  4.9G   2% /media/user
/dev/sdb7     1009M  869M  132M  87% /media/rootfs
```

- **Now, we need to make Tizen specific changes in raspberry pi kernel and Tizen platform to boot the image successfully on Pi2.**

TIZEN Kernel Customization

- **Download Raspberry Pi Kernel (4.1.16) repo from the following:**
  - git clone --depth=1 git://github.com/**raspberrypi**/linux
- **Build the kernel:**
  - make ARCH=arm -j8 CROSS_COMPILE=arm-linux-gnueabi-**bcm2709_defconfig**
  - make ARCH=arm -j8 CROSS_COMPILE=arm-linux-gnueabi-
- **Create a new defconfig for Tizen:**
  - # cp –f .config arch/arm/configs/**tizen_pi2_defconfig**
- **Enable Tizen specific kernel configurations (one by one), using:**
  - make ARCH=arm menuconfig
- **Each time you change the configuration, make sure to sync with the tizen_pi2_defconfig.**

- **Fortunately, in Raspberry Pi Kernel (bcm2709_defconfig), most of the Tizen configs are already enabled.**
- **However, we still need to enable few once as below:**

CONFIG_SECURITYFS=y
CONFIG_SECURITY_SMACK=y
CONFIG_AUDIT=y
~~CONFIG_DRM=y~~
CONFIG_MEMCG=y

CONFIG_MEMCG_SWAP=y
CONFIG_ZRAM=y
CONFIG_CGROUP_DEBUG=y
CONFIG_PM_SLEEP=y
CONFIG_PM_AUTOSLEEP=y

- **After enabling these configs, make sure to sync the .config with the default tizen_pi2_defconfig again.**

- **Build the final Kernel image:**
  - make ARCH=arm -j8 CROSS_COMPILE=arm-linux-gnueabi-**tizen_pi2_defconfig**
  - make ARCH=arm -j8 CROSS_COMPILE=arm-linux-gnueabi-
- **Generate the device tree image:**
  - ./scripts/**mkknlimg** arch/arm/boot/zImage **kernel7.img**
- **Copy the kernel images to the SD card boot partition.**
  - cp -f **kernel7.img** /media/boot/
  - cp -f arch/arm/boot/dts/**bcm2709-rpi-2-b.dtb** /media/boot/
  - cp -f arch/arm/boot/dts/**overlays**/*.dtb /media/boot/overlays/

- **Install the modules built by kernel:**
  - make ARCH=arm CROSS_COMPILE=arm-linux-gnueabi- -j8 INSTALL_MOD_PATH=**../modules** modules_install
- **Copy the modules & firmware to SD card rootfs folder:**
  - sudo cp -rf modules/lib/modules/**4.1.16-v7+** /media/**rootfs**/lib/modules/
  - sudo cp -rf modules/lib/**firmware**/* /media/**rootfs**/lib/firmware/
- **Copy the original pi2 firmware from NOOBs root folder to the SD card Tizen rootfs folder:**
  - sudo cp -rf <pi2 noobs root>/lib/firmware/* /media/rootfs/lib/firmware/

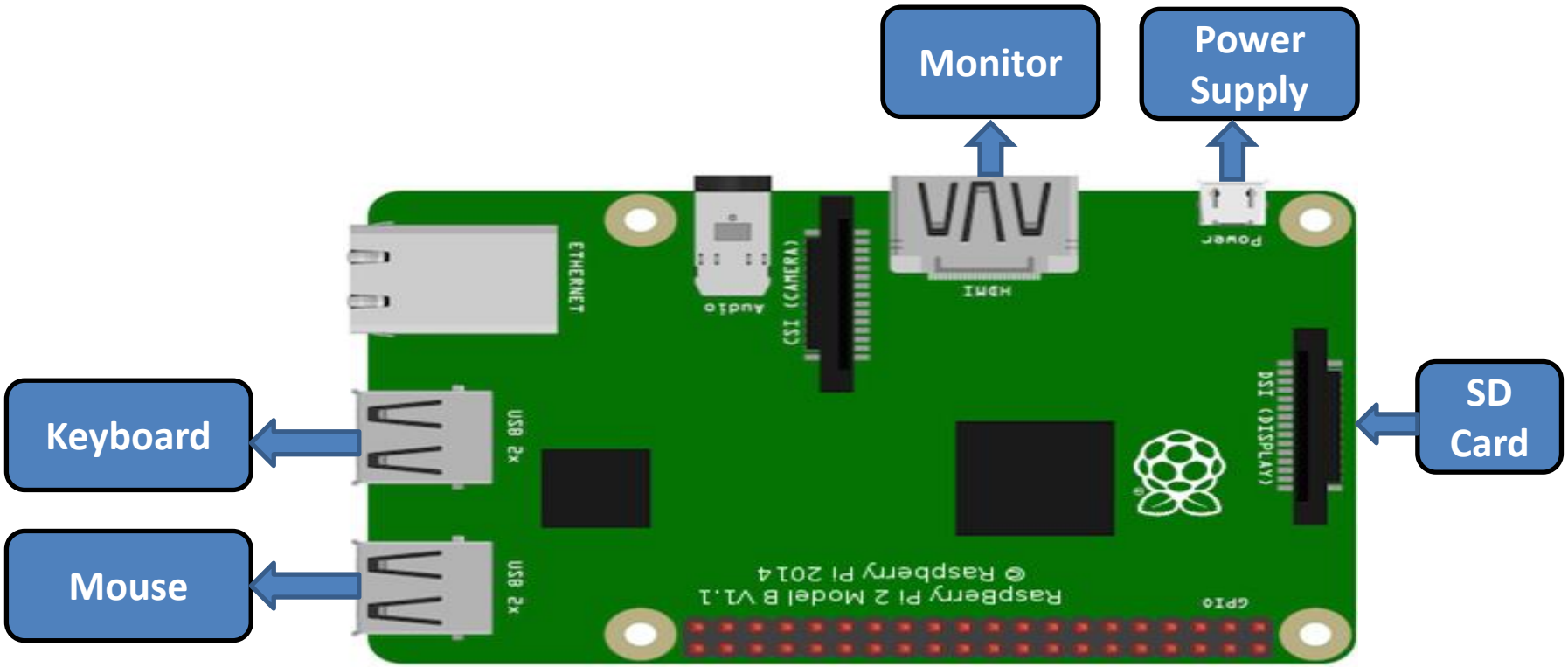
TIZEN Platform Customization

- **Update rootfs device node under Kernel command line in SD card partition: /media/boot/cmdline.txt**
  - dwc_otg.lpm_enable=0 console=ttyAMA0,115200 console=tty1 root=**/dev/mmcblk0p7** rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait
- **Tizen uses systemd services for booting, so customize systemd services as per your needs, under:**
  - **/usr/lib/systemd/system/***
- **For example to boot the system till command prompt, you can set the default.target to multi-user.target**
- **At this time, you can also perform the clean up of unnecessary services by simply deleting it and removing it dependencies.**

- **In Tizen, the getty and console services are disabled by default.**
- **We need to enable these services to get the login prompt on the terminal.**
- **To enable these services we need to modify the following file:**
  - /media/rootfs/usr/lib/systemd/system-preset/**90-systemd.preset**
    - **enable console-getty.service**
    - **enable console-shell.service**
- **To get login prompt on Virtual Terminal (tty1), we need to create tty1 service file:**
  - cd /media/rootfs/usr/lib/systemd/system/multi-user.target.wants
  - sudo ln –s ../getty@.service **getty@tty1.service**

SETUP & INTERFACING

- Plug the SD card on the PI2 hardware.
- Plug other required peripherals as shown below and power on the raspberry pi.



- You will be able to see the console messages flowing on the monitor screen.
- If Tizen platform images are mounted successfully, you will be able to see the following on the terminal.

```
Detected architecture 'arm'.

Welcome to Tizen 3.0.0 (Tizen3/Common)!

No hostname configured.
Set hostname to <localhost>.

localhost login:
```

```
Welcome to Tizen
root@localhost:~#
```

User name: {root, guest} Password: tizen

Note:
User name and password can be found from .ks file.
tizen-common_20160315.2_common-wayland-3parts-armv7l-odroidu3.ks

Camera Setup

- In raspberry pi, camera interfacing can be done in 2 ways:
  - Using CSI Camera Slot
  - Using the USB Web Cam

- CSI Camera:
  - In CSI camera slot we can directly plug a raspberry pi 5MP Camera module using the ribbon cable.
  - It is directly controlled by GPU and it is faster. But it requires around 128MB of system RAM reserved memory.
  - However, we can convert this memory to CMA if memory saving is important.
  - https://www.raspberrypi.org/documentation/usage/camera/README.md

- **USB Web Camera:**
  - In one of the USB slot plug a webcam (Logitech Webcam)
  - It will create a node : /dev/video0 through which we can access it using V4L2 calls.
  - It does not need any reserved memory but the processing could be little slower compared to CSI camera.
  - Using USB you can connect any number of web cams.
  - https://www.raspberrypi.org/documentation/usage/webcams/

- **On Tizen, we can perform camera capture using the following:**
  - **Using the standard Gstreamer commands**
  - **Using a simple V4L2 application**
  - **Using the mm_test_suite for the Tizen source code repo.**
  - **Using launch_cam.sh (only for web cam) [For Tizen 3.0]**

- **Gstreamer command for single frame capture:**
  - gst-launch-1.0 **v4l2src** num-buffers=1 ! video/x-raw, format=I420, width=640, height=480, framerate=30/1 ! filesink location=**/opt/usr/media/file.yuv**
- **Other sample applications are available under Tizen source:**
  - https://review.tizen.org/git/?p=platform/core/api/camera.git;a=tree
- **MM Camcoder test suite is available under:**
  - https://review.tizen.org/git/?p=framework/multimedia/libmm-camcorder.git;a=tree;f=test;h=49ee5fc53d4fcb99e37a1cd5c554f0c95974f365;hb=HEAD

Display/Graphics Setup

- Tizen uses DRM (Direct Rendering Manager) & X11/Wayland based display system.
- Both DSI display connector or HDMI display interface can be used.
- The DRM support for Raspberry Pi graphics controller VC4 is already available from Linux Kernel 4.1.16.
  - Linux/driver/gpu/drm/vc4/…
- Tizen graphics port for Raspberry Pi is already available as part of Tizen Yocto project repo for PI2.
  - https://blogs.s-osg.org/tizen-rpi2-now-supporting-3d-acceleration/

Bluetooth Setup

- **To setup Bluetooth on Tizen, insert the Bluetooth USB Dongle and perform the following steps:**
  - **a)** **root@localhost:~#** hciconfig hci0 up
  - **b)** **root@localhost:~#** **bluetoothctl**
  - **c)** **[bluetooth]#** power on
  - **d)** **[bluetooth]#** agent on
  - **e)** **[bluetooth]#** scan on
  - **f)** **[bluetooth]#** pair <scanned device MAC_ID>
  - **g)** **[bluetooth]#** connect <MAC_ID>
  - **h)** **[bluetooth]#** exit

For more information please visit:

https://wiki.tizen.org/wiki/Connecting_to_a_Smartphone_with_Bluetooth_and_Making_Phone_Calls

Wi-Fi Setup

- **In Tizen 3.0, we can configure Wi-Fi from the command prompt using the following steps:**

a) **root@localhost:~#** ifconfig wlan0 up

b) **root@localhost:~#** wpa_supplicant –u –t –B –d –Dwext –f/var/log/wpa_supplicant.log

c) **root@localhost:~#** **connmanctl**

d) **connmanctl>** enable wifi

e) **connmanctl>** agent on

f) **connmanctl>** services

- wifi_<wlan0_MAC_ID>_<XXXXX>_managed_psk

g) **connmanctl>** connect wifi_<XXXXX>_managed_psk

[Enter the passphrase here] xxxxxxxx

h) **connmanctl>** exit

- **If Wi-Fi is connected properly, an IP Address would be assigned to wlan0 interface:**

```
root@localhost~:# ifconfig
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu
1500
        inet 192.168.43.91  netmask 255.255.255.0  broadcast
192.168.43.255
        inet6 fe80::2c1:41ff:fe29:9c80  prefixlen 64  scopeid 0x20<link>
        ether 00:c1:41:29:9c:80  txqueuelen 1000  (Ethernet)
        RX packets 24  bytes 2789 (2.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 37  bytes 5470 (5.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

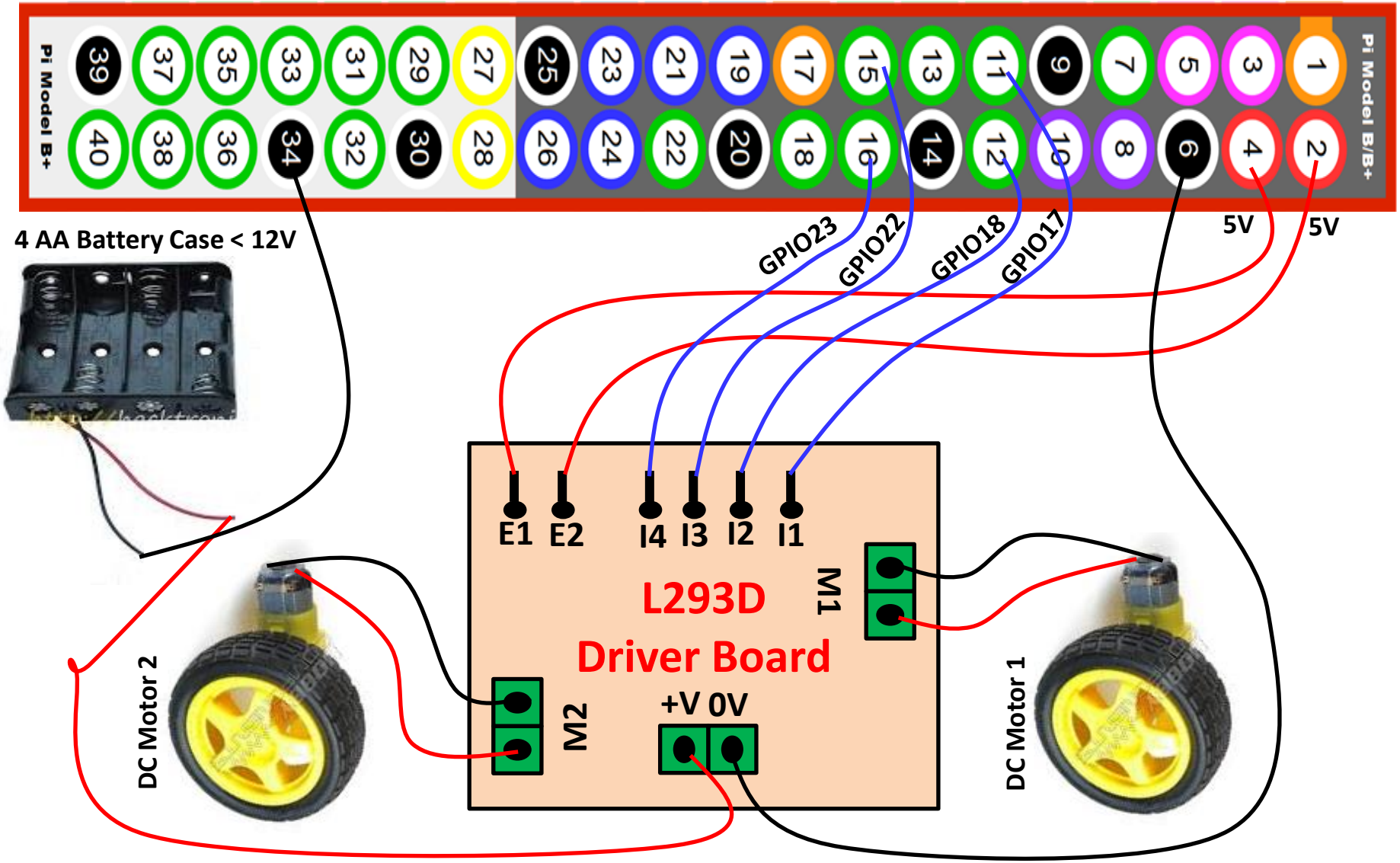
https://blogs.s-osg.org/setup-wifi-raspberry-pi-2-tizen/

HARDWARE INTERFACING

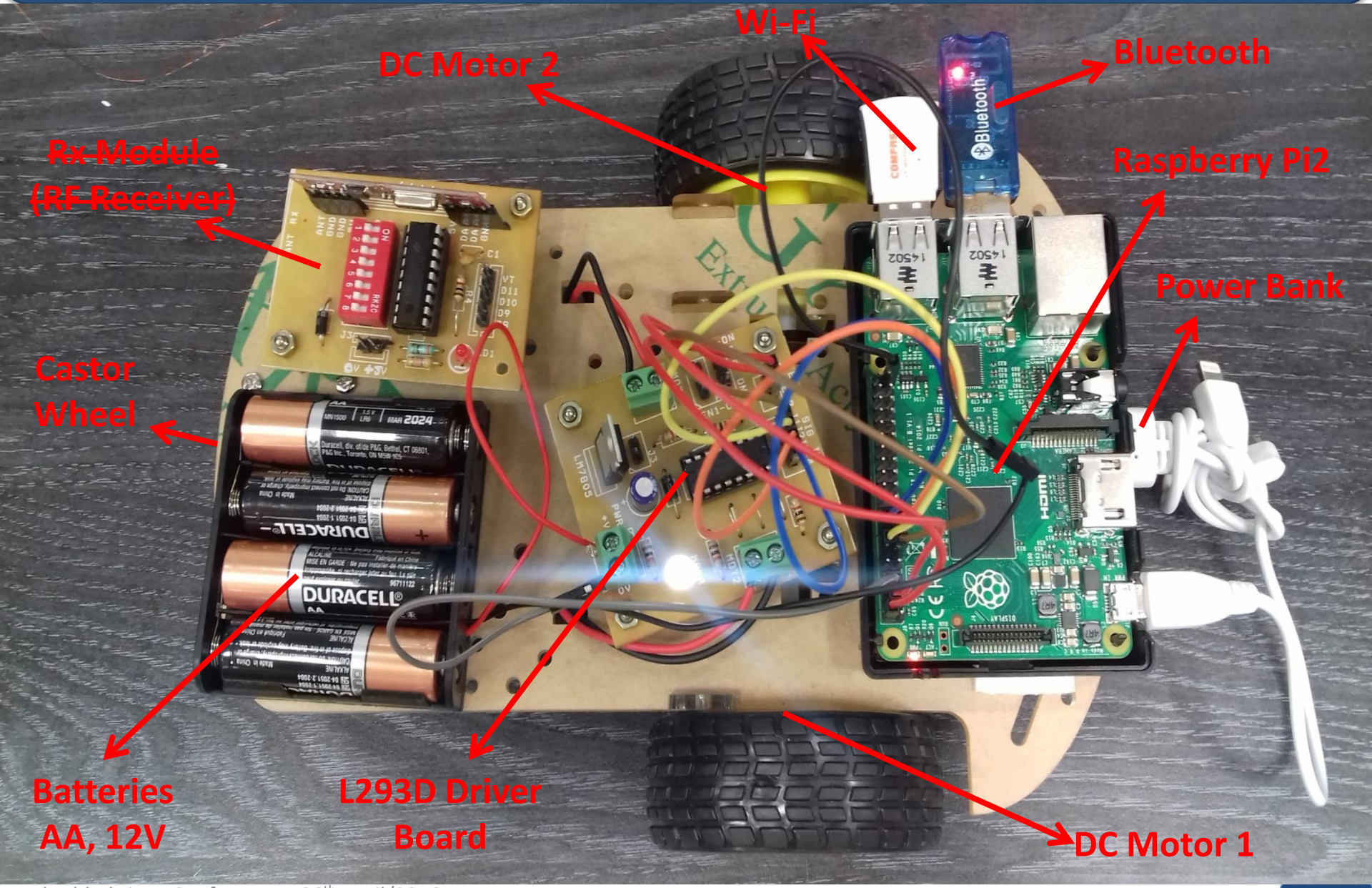
- Assemble the Robot DIY kit (that includes: chassis, DC motors, rubber wheels, castor wheels).
- Attach the Raspberry Pi2 to the chassis.
- Attach the power bank under the chassis and connect the USB cable to it (Do not connect to RPi2 now).
- Attach the L293D driver board to the chassis.
- Stick the battery case on the chassis (Do not put the battery).
- Connect the motor driver as shown in the next slide.
- Connect the Wi-Fi, Bluetooth, Webcam to the RPi2 USB slot.
- Temporarily connect the keyboard and monitor to do the initial configuration and setup (Remove it once done).
- Now, connect the power bank USB cable to the RPi2 (to power on and boot the Pi2).
- Finally, connect the battery to the battery case.

Motor Connection with PI2

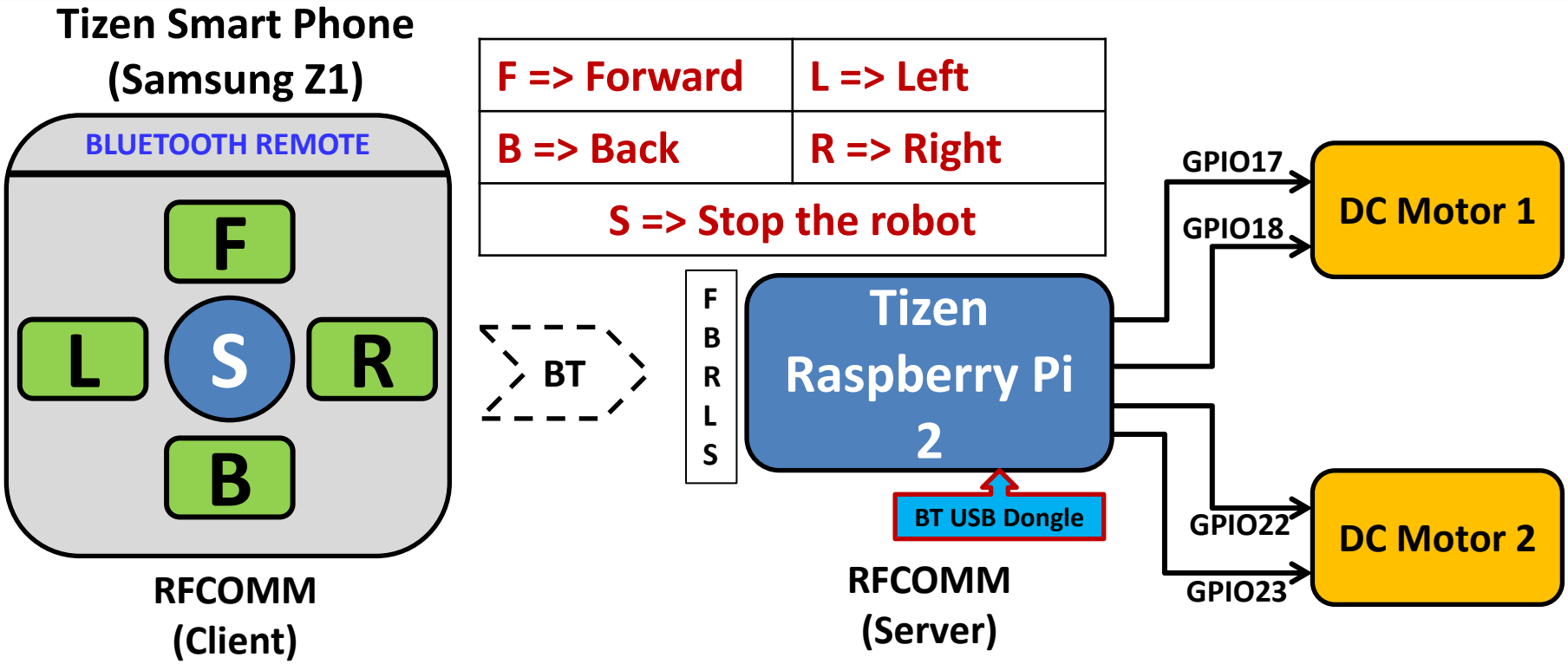
Raspberry Pi 2 - > GPIO Pins



Full Model (Tizen Inside)



ROBOT CONTROL MECHANISM



Forward (F)	Back (B)	Left (L)	Right (R)	Stop (S)
GPIO17: 1	GPIO17: 0	GPIO17: 0	GPIO17: 1	GPIO17: 0
GPIO18: 0	GPIO18: 1	GPIO18: 0	GPIO18: 0	GPIO18: 0
GPIO22: 1	GPIO22: 0	GPIO22: 1	GPIO22: 0	GPIO22: 0
GPIO23: 0	GPIO23: 1	GPIO23: 0	GPIO23: 0	GPIO23: 0

- The devices should be already paired and connected.
- Use Tizen Mobile 2.4 SDK to develop RFCOMM Client App.
  - Reference: https://developer.tizen.org/dev-guide/2.4/org.tizen.native.mobile.apireference/group__CAPI__NETWORK__BLUETOOTH__SOCKET__MODULE.html
- Use Tizen CAPI to develop RFCOMM server that runs as a Daemon to receives data from client and take action.
  - Reference: https://review.tizen.org/git/?p=framework/api/bluetooth.git;a=tree;f=test;h=e7732ccffdc87b0ae64c55e5486581a4b5956653;hb=HEAD
- To control the motor, we can simple write {1,0} to the respective GPIOs as shown in the table, using the GPIO sysfs entries.
  - echo 1 > /sys/class/gpio/gpio17/value
  - echo 0 > /sys/class/gpio/gpio18/value
  - echo 1 > /sys/class/gpio/gpio22/value
  - echo 0 > /sys/class/gpio/gpio23/value

Forward

RAM Usage

- RAM memory usage just after boot-up with Wi-Fi, Bluetooth connected (without display)

 # free -tm

	total	used	free	shared	buffers	cached
Mem:	973	137	835	12	6	75
-/+ buffers/cache:			56	916		
Swap:	255	0	255			
Total:	1229	137	1091			

RAM: 1GB (1024 MB)
Reserved Memory: (1024 – 973) = 51 MB
Used during boot-up: 137 MB
Total Used: 51 + 137 = 188 MB
Swap (ZRAM) = 256 MB (1/4th for 1GB)
[Change zram size in: /etc/resourced/swap.conf]

Note:
To get process wise memory usage, we can use a special command in Tizen:
 # memps –a

Reserved Memory Details

Kernel Code Size:

 # size –t vmlinux

text	data	bss	dec	hex	filename
8314042	690396	784004	9788442	955c1a	vmlinux
8314042	690396	784004	**9788442**	955c1a	(TOTALS)

 # du –h modules

48M modules

Kernel Reserved:

 # dmesg | grep –i memory

Memory: 988016K/1015808K available (6123K kernel code, 527K rwdata, 1688K rodata, 448K init, 757K bss, 19600K reserved, 8192K cma-reserved)

Total RAM visible to Kernel = 1015808K = 992MB

Reserved for GPU = 16MB (cat /media/boot/config.txt : gpu_mem=16)

Reserved memory others = 16MB (????)

Kernel Reserved = 19600 = 19.14MB (includes kernel code & data structures)

- **Kernel code size can be reduced below 5MB.**
- **Platform memory can be optimized further by analyzing memps report.**

ROM Details

- ROM memory details with Tizen common 3.0 profile.

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/root	945M	806M	114M	88%	/
devtmpfs	483M	4.0K	483M	1%	/dev
tmpfs	487M	4.0K	487M	1%	/dev/shm
tmpfs	487M	13M	475M	3%	/run
tmpfs	487M	0	487M	0%	/sys/fs/cgroup
tmpfs	487M	8.0K	487M	1%	/tmp
/dev/mmcblk0p8	180M	384K	175M	1%	/opt
/dev/mmcblk0p9	4.9G	14M	4.8G	1%	/opt/usr
tmpfs	487M	0	487M	0%	/opt/usr/share/crash/temp
tmpfs	98M	0	98M	0%	/run/user/5001

- Rootfs size is less than 1GB. Further reduction is possible.
- Still we have lots of space in usr partition.
- For our use case, even 2GB storage should be enough.
- User files can be stored in /usr/share/media/ folders.

Loaded Modules (lsmod)

Module	Size	Used by
rfcomm	33992	2
btusb	29353	0
bnep	10479	2
btintel	1369	1 btusb
btbcm	4490	1 btusb
xt_connmark	1735	0
iptable_nat	1646	0
nf_conntrack_ipv4	13237	1
nf_defrag_ipv4	1321	1 nf_conntrack_ipv4
nf_nat_ipv4	4891	1 iptable_nat
nf_nat	12207	1 nf_nat_ipv4
nf_conntrack	76946	4 nf_nat,nf_nat_ipv4,xt_connmark,nf_conntrack_ipv4
xt_mark	998	0
iptable_filter	1275	0
iptable_mangle	1379	1
ip_tables	11439	3 iptable_filter,iptable_mangle,iptable_nat
x_tables	13353	5 xt_mark,ip_tables,iptable_filter,xt_connmark,iptable_mangle
bluetooth	324803	26 bnep,btbcm,btusb,rfcomm,btintel
bcm2835_gpiomem	2973	0
bcm2835_rng	1770	0
arc4	1778	2
rt2800usb	17476	0
rt2800lib	71877	1 rt2800usb
crc_ccitt	1149	1 rt2800lib
rt2x00usb	8539	1 rt2800usb
rt2x00lib	36483	3 rt2x00usb,rt2800lib,rt2800usb
mac80211	523380	3 rt2x00lib,rt2x00usb,rt2800lib
snd_bcm2835	19620	0
snd_pcm	74535	1 snd_bcm2835
snd_timer	18419	1 snd_pcm
snd	52151	3 snd_bcm2835,snd_timer,snd_pcm
cfg80211	403784	2 mac80211,rt2x00lib
uio_pdrv_genirq	2997	0
uio	7880	1 uio_pdrv_genirq
rfkill	16398	4 cfg80211,bluetooth
joydev	9213	0
evdev	10421	5
sch_fq_codel	7858	2
ipv6	341361	20

BENEFITS OF USING TIZEN

- Tizen is truly a open source platform. Every component used in Tizen is derived from open source. So we have huge flexibility to customize as per our needs.
- Tizen uses profile concept to support new devices. So new use case can be easily derived using one of the profile.
- As we have seen, it is very easy to create new profile to support future technologies.
- With Tizen is easy to create bare minimal functionalities with lesser foot prints.
- Because of it's multi-device capabilities it is possible to create device convergence and derive new communication mechanism.
- Finally, using the power of open h/w and open source OS, it is easy to perform various experiments before deriving actual products.

FUTURE WORK

- Perform various clean-ups and create a simple Robotics profile.
- Touch screen display bring-up using DSI connector.
- CSI camera bring-up.
- SDB bring-up and integration in Raspberry Pi Kernel.
- Various sensors interfacing with the robot.
- Power consumption analysis while robot is in operation.
- Setting up the web server and controlling robot using Wi-Fi.
- Getting camera preview remotely on smart phone.
- Contribute all the changes to upstream and update in Tizen wiki.

- Others who like to contribute can join:
  - https://wiki.tizen.org/wiki/How_to_contribute_to_Tizen_on_Yocto_Project
- Community:
  - https://www.tizen.org/community/mailing-lists

REFERENCES

- https://www.raspberrypi.org/products/raspberry-pi-2-model-b/
- https://wiki.tizen.org/wiki/Porting_Guide#
- https://wiki.tizen.org/w/images/8/86/LinuxCon14_TizenCommon_20141015.pdf
- https://www.tizen.org/ko?langredirect=1
- https://review.tizen.org/git/
- https://wiki.tizen.org/wiki/Tizen_on_Yocto_Project
- http://blogs.s-osg.org/category/tizen/
- https://blogs.s-osg.org/tizen-on-rpi2/
- http://events.linuxfoundation.org/sites/events/files/slides/KLF2014-Dongkun.pdf
- http://www.krnet.or.kr/board/data/dprogram/1784/C1-2-KRnet2013.pdf
- https://people.csail.mit.edu/albert/bluez-intro/
- http://diyhacking.com/diy-projects/raspberry-pi-projects/
- http://diyhacking.com/raspberry-pi-robot/

THANKS